Summary

"Read It" is a program that generates your choice of either sentences or mad libs. The program is meant to assist young children as they learn to read. The program allows children to choose grade level and difficulty of the sentences generated. Characteristic of the original mad libs, all sentences generated are usually comical in nature but are grammatically correct. Since the purpose of the program is to introduce children to new words, there is a "Say it" option so that a child may hear the sentences if they are having trouble reading a particular word or phrase.

Implementation

- The key to "Read It" is the database stored in the excel file. Originally, the program was going to accept input from the user. The input would have been a word document or a website. The program would then compile all of the words from the document or website and generate sentences based on the compilation. This would have required a web dictionary search for each and every word in the site or document. Also, this would not be user friendly—especially not for children. Each sheet of the excel file contains hundreds of words from the different parts of speech of the English language (noun, verb, adverb, adjective, noun, preposition, pronoun).
- It was important to be able to adjust certain words to fit the right sentence. I wrote code that reads in a value for adjectives and converts them to superlative form (ending in "est"). The other part of speech that required transformation is nouns. Though most of the parts of speech can be found at "momswhothink.com", I needed to create a new sheet and individually alter each of the nouns to fit their plural form.
- I wrote several formulas (one for every part of speech) that would automatically return a random word. For example, the Get_noun function returns a random noun based on a selection from the user.
- The form has several categories to choose from. The user can choose different selections in the following categories: "grade level", "adjective category", or "verbs that start with:", or the user may choose "any" for each of these categories. Within each of the functions for parts of speech I included variables that were randomized based on how many selections there are in each category. Since the sheets are all organized in categories, code had to be added to the functions for when the user selects "any". It first randomizes the category and then randomizes the word.
- For the sentence generator, I specified word order for each difficulty. The "Beginner" difficulty displays a short sentence with little more than a subject and a noun. The "Average" difficulty incorporates multiple nouns and verbs. The "Advanced" difficulty adds the use of adverbs.
- For the mad lib generator the user will use the "Name" field to personalize their mad lib. The mad libs account for gender, which can be chosen by the user. Formatting the mad lib generator was quite time consuming since each phrase (in some cases each word) had to be placed in a different cell of the

"Libs" worksheet. The proper amount of spaces had to be used before and after the words that were generated. The stories are fixed but many of the pieces of the story are randomized which gives a comical spin on the stories. This was the section where I had to convert adjectives to their superlative form. The code for this was written using several "If/Then" statements to account for the rules of English superlatives.

Discussion

I think the most important thing I learned was the importance of simplicity in creating a program that is bearable to use. When I wrote the original code for this project it was clunky and non-user-friendly. Because of having to turn to the Internet for every single word definition, the program spent over a minute to configure just one sentence. At this point in the process the program could serve no real purpose. Since it originally took its data from a user-specified website or document, there was no way to screen the words that were read in. It was a breakthrough to realize that this program could be very useful for children in helping them read if I could screen for elementary level words. Finding that there were resources that provide thousands of words specifically for elementary school students, I realized that a slow, laggy, Internet search was unnecessary. Now the program runs quickly and is customized to fit the vocabulary that elementary school children should be learning.

I wanted to include a small thumbnail photo that would appear near the bottom of the form. The photo was supposed to be a visual help that would display a picture based on what noun had been generated in the sentence generator. At first, I wanted to implement the Internet search to find a picture based on search criteria (the noun) and display it in the form. Unfortunately, since this program is directed towards children, I would have had to find some way to screen the photos so that no inappropriate photos were displayed. I actually spent several hours downloading "child friendly" jpg photos for each of the nouns in the "Kindergarten" and "1st Grade" sections. In the end, I decided that the picture would not serve any better purpose than the "Say It" feature. The photos would have made the form look too tacky. In the future, I may come up with a classier version of this idea.